TIEGCM Documentation

Release 1.94

Ben Foster

May 17, 2011

CONTENTS

Note: This document is up to date for TIEGCM version 1.94

Contents:

INTRODUCTION

The NCAR/HAO Thermosphere Ionosphere Electrodynamics General Circulation Model, or TIEGCM, is a three-dimensional, time-dependent, numeric simulation model of the Earth's upper atmosphere, including the upper Stratosphere, Mesosphere and Thermosphere.

This document is intended to assist members of the upper atmosphere research community in building, executing, and visualizing the TIEGCM as an integral part of their research. We also welcome development efforts on the model itself, subject to the Academic Research License Agreement `tiegcmlicense.txt`.

For more information about the TIEGCM and other related models, please see our website at
http://www.hao.ucar.edu/modeling/tgcm/

QUICKSTART PROCEDURE

This document is intended to provide a tutorial-like procedure for downloading source code and start-up data files, building the model, and executing a short default run on a 64-bit Linux system.

Note: This document is up to date for version 1.94 of the TIEGCM

2.1 Downloading the model source code and required data files

The model source code and related input data files may be downloaded from the TIEGCM download page of the main TGCM website:

http://www.hao.ucar.edu/modeling/tgcm/download.php

You will need to provide an email address (login and password are NOT required). Documentation and Postprocessor codes are also available on the download site, but all you need for now is the source code, and corresponding data files. Both are provided as gzipped tar files.

After downloading the tar files to an empty working directory (*workdir*) on a large disk system (for example, /mydisk/tiegcm), uncompress and extract the source code and related scripts and documentation:

```
$ gunzip tiegcmx.xx.tar.gz
$ tar xvf tiegcmx.xx.tar
```

where x.xx is the model version downloaded. Next, make a directory to hold the data files (for example, /mydisk/tiegcm/data), and uncompress and extract the data tar file into that directory. Then set environment variable $TGCMDATA, e.g., for the c-shell, add this line to your .cshrc file:

```
setenv TGCMDATA /mydisk/tiegcm/data
```

At this point, you should have something like this in your working directory /mydisk/tiegcm:

```
total 2376
-rw-rw-r-- 1 user tgcm     4928 Jun  1  2010 README
-rw-r--r-- 1 user tgcm     2886 Jun  1  2010 Release_Notes
drwxrwxr-x 2 user tgcm     4096 Apr 22 08:52 data/
-rwxrwxr-x 1 user tgcm    10137 May 31  2010 tiegcm-ibm.job*
-rwxrwxr-x 1 user tgcm    10671 May 31  2010 tiegcm-linux.job*
drwxrwxr-x 5 user tgcm     4096 Jun  1  2010 tiegcmx.xx/
-rw-r--r-- 1 user tgcm     6116 May 31  2010 tiegcmlicense.txt
```

These files and directories contain the following:

README
> Instructions for building and making a short default run.

Release_Notes
> Release notes for this version of the model.

data/
> Directory containing the downloaded data files (this is `$TGCMDATA`)

tiegcm-ibm.job
> Job script for building and executing under IBM/AIX systems. (default `tiegcm-ibm.job`)
>
> Read more about *Running the model on IBM/AIX systems*.

tiegcm-linux.job
> Job script for building and executing under Linux (64-bit) systems. (default `tiegcm-linux.job`)

tiegcmx.xx/
> Model root directory, containing source code, supporting scripts, and documentation.

You are now prepared to build the model and make a short default run using the job script.

2.2 Making a default run on a 64-bit Linux system

Take a look at the Linux job script `tiegcm-linux.job`. Near the top are several shell variables, with their default settings, which configure the job script (variables and values may vary somewhat between model versions):

```
set modeldir = tiegcmx.xx
set execdir  = tiegcm-linux
 set make      = Make.intel_hao64
#set make      = Make.pgi_hao64
#set input     = tiegcm.inp
set output   = tiegcm.out
set mpi       = TRUE
set nproc     = 4
set modelres = 5.0
set debug     = FALSE
set exec      = TRUE
set utildir  = $modeldir/scripts
```

Following are brief explanations of the job script shell variables:

Note: Absolute or relative paths are acceptable when specifying directories. Relative paths should be relative to the *working directory* (*workdir*).

---

modeldir
> The model root directory (*modeldir* from the source code download). This will contain subdirectories `src/`, `scripts/`, `doc/`, etc.

tiegcm-linux
> This is the execution directory (*execdir*), in which the model will be built and executed. It will be created if it does not already exist. This directory will also contain the model output netCDF history files.

make
> Make file containing platform-specific compiler flags, library locations, etc. If not otherwise specified with a path, the job script will look for this file in `modeldir/scripts`. This file is included in the main Makefile (`scripts/Makefile`). The user can either make necessary adjustments to an existing `make` file, or write their own for a different platform/compiler system.

Here is an example `make` file for 64-bit HAO Linux systems using the ifort Intel compiler: `Make.intel_hao64`

input

The namelist input file. When this is commented (as above), the job script will make a default namelist file `tiegcm_default.inp`, and use it for the default run. Later, you can edit this file for your own runs, rename it, and reset and uncomment the `input` shell variable in the job script.

output

Name of the file to receive stdout output from the model. If this pre-exists, it will be overwritten when the model is executed.

Here is an example stdout file from a single-processor default run: `tiegcm_default.out`

mpi

Logical flag indicating whether or not to link the MPI library for a multi-processor parallel run. If FALSE, the MPI library is not linked, and it is assumed the model will be run in serial (single-processor) mode.

nproc

Number of processors to use in a parallel execution. This is ignored if `mpi` is FALSE.

modelres

Model resolution. Two resolutions are supported:

- modelres = 5.0 sets 5-degree lat x lon horizontal, and dz=0.50 vertical
- modelres = 2.5 sets 2.5-degree lat x lon horizontal, and dz=0.25 vertical

If the resolution is changed, the model should be recompiled before re-executing the job script (type "*gmake clean*" in the execdir)

debug

If `debug` = TRUE, the job script will compile the build with debug flags set. Debug flags specific to the compiler are set in the `make` file. If `debug` is changed, the code should be recompiled (type "gmake clean" in the `execdir` before re-executing the job script).

exec

If `exec` = TRUE, the job script will execute the model after compilation, otherwise, the job script will stop after compilation without execution.

utildir

The utility directory containing supporting scripts. This is normally the `scripts/` subdirectory in the model root directory `modeldir`.

You are now ready to build and execute a default run. To do this, simply execute the job script as follows:

```
$ tiegcm-linux.job &
```

The compilation output will be displayed. If the build is successful (and exec=TRUE), the model will be executed, and stdout will go to the specified `output` file. If the job is successful, you can edit and rename the namelist input file, reset `input` in the job script, and re-execute the job script. If there has been no change to the source code, it will not need to recompile, and will use the pre-existing executable.

2.3 Running the model on IBM/AIX Platforms

Note: This section contains some information that is specific to user's of the NCAR IBM system `"bluefire"`. User's of other IBM systems may need to make adjustments for their particular environment. For more information about the NCAR bluefire system, see http://www2.cisl.ucar.edu/docs/bluefire-user-guide

The model can be built and executed on IBM platforms running AIX with the xlf90 (mpxlf_r) compiler. You can use the same procedure described in the previous section, except that you use the IBM job script `tiegcm-ibm.job` instead of the Linux job script tiegcm-linux.job.

The IBM job script has the same user-settable shell variables as the Linux job script, but the default settings are slightly different:

```
set modeldir = tiegcm_trunk
set execdir  = tiegcm_trunk-aix
#set input     = tiegcm.inp
set output   = tiegcm.out
set make     = Make.bluefire
set mpi      = TRUE
set modelres = 5.0
set debug    = FALSE
set exec     = TRUE
set utildir  = $modeldir/scripts
```

Note the `execdir` name, and the `make` file `Make.bluefire`

Also note the special "`#BSUB`" directives at the top of the IBM job script (descriptions in the right-hand column are for this document only, and are not in the script itself):

```
#BSUB -J tiegcm_trunk              # Job name
#BSUB -P 24100004                  # NCAR project number
##BSUB -q regular                  # regular queue (commented here)
##BSUB -n 32                       # number of processors (commented here)
#BSUB -q debug                     # debug queue
#BSUB -n 8                         # number of processors (MPI tasks)
#BSUB -o tiegcm_trunk.%J.out       # stdout file
#BSUB -e tiegcm_trunk.%J.out       # stderr file
#BSUB -N
#BSUB -u $LOGNAME@ucar.edu         # email notification address
#BSUB -W 1:00                      # wallclock limit (6-hr max at NCAR)
```

These are resource settings for the Load Sharing Facility (LSF), the batch queuing system sold by Platform Computing. The LSF is used for scheduling jobs on the `bluefire` IBM system at NCAR. This job will be submitted to the debug queue, requesting 8 processors, with a wallclock limit of 1 hour. Note the double pound-sign "##" indicates a commented field.

To submit the IBM job script to the LSF batch system, type:

```
$ bsub < tiegcm-ibm.job
```

Watch the progress of your LSF job with the command:

```
$ bjobs
```

You can kill a LSF job with this command:

```
$ bkill job_ID
```

Where `job_ID` is the job identifier given in the `bjobs` command.

For more information about the LSF, see the Wikipedia site:

http://en.wikipedia.org/wiki/Platform_LSF

or the Platform Computing site:

http://www.platform.com/workload-management/high-performance-computing/lp
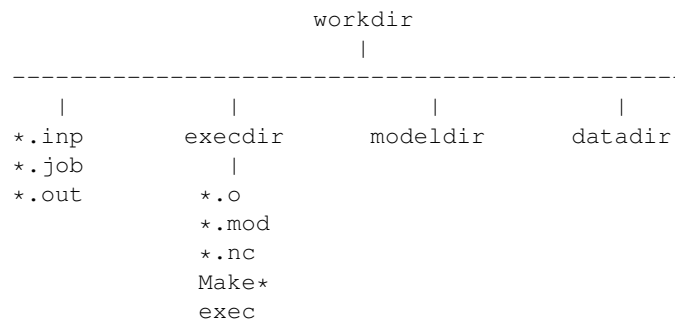
DIRECTORY STRUCTURE

This section describes a typical directory structure the user will be working with when running the TIEGCM model. The working directory (*workdir*) is the "root" directory of the user's project.

The model directory (*modeldir*) is typically a subdirectory under the working directory, and contains the model source code, supporting scripts, documentation, and scripts for running tests.

The data directory (*datadir*) may also be a subdirectory under the working directory, or it may be on a large temporary disk that is accessible from the working directory. The data directory contains start-up and input data files for running the model.

3.1 Working Directory (*workdir*)

The user's working directory will typically look something like this:

```
                    workdir
                       |
_____
    |            |             |            |
*.inp       execdir      modeldir      datadir
*.job          |
*.out        *.o
             *.mod
             *.nc
             Make*
             exec
```
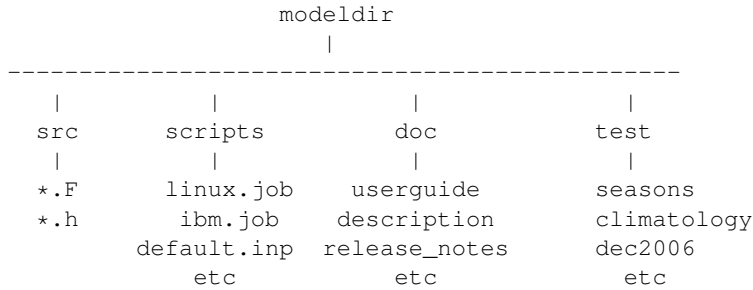
Where *.inp are namelist input files, *.job are job scripts, and *.out are stdout files from model runs. The *execdir* is the build/execution directory (created by the first model run), with object code (*.o, *.mod), model output history files (*.nc), make files (Make*), and an executable file. Various other files may also be in the execdir. The modeldir and datadir directories are described below.

The job script in your working directory contains a shell variable specifying the modeldir, so it knows where to find the source code and supporting scripts for the build process. The namelist input file refers to the datadir for start-up and other data input files (e.g., SOURCE, GPI_NCFILE, IMF_NCFILE, etc). These namelist parameters can use the environment variable TGCMDATA to specify the datadir (see section on *namelist input files*).

3.2 Model Directory (*modeldir*)

The model root directory is what you get when you *download* the model source code tar file. The model directory contains subdirectories with the model source code, supporting scripts, documentation, and test scripts:

```
                modeldir
                   |
---------------------------------------------
    |          |            |            |
   src      scripts        doc          test
    |          |            |            |
  *.F       linux.job    userguide     seasons
  *.h        ibm.job     description   climatology
           default.inp  release_notes  dec2006
               etc          etc           etc
```

The *src/* directory contains all source code, i.e., all *.F fortran files, and a single header file with grid definitions and dimensions (defs.h). The TIEGCM source code is f90 compliant, and most files are written in fixed-format Fortran.

The *scripts/* directory contains various csh, perl and other scripts providing infrastructure support for the model. These include several Make files, sample job scripts, a default namelist input file, and several utility scripts.
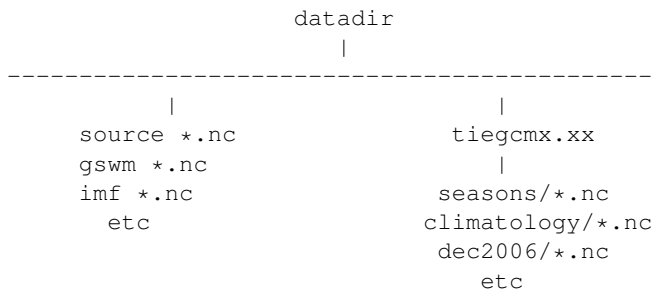
The documentation directory *doc/* contains the User's Guide (this document), and Model Description (pdf and html files). These docs are also available in the documentation section of the main tgcm website tgcm website. The User's Guide is written in Restructured Text, with the Sphinx markup tool. Source code (.rst and other files) for the User's Guide are available in the "userguide" subdirectory. The Model Description is written in TeX, and its source code is available in the "description" subdirectory.

Also in the doc directory are Release Notes for the current version, a README file, and other relevant documents (e.g., table of available diagnostic fields from the model).

The *test/* directory contains setup scripts for various test/benchmark runs, (seasonal, full-year climatology, storm cases, etc.) Each script produces job scripts and input files to make the runs for that test.

3.3 Data Directory (*datadir*)

The initial data directory is what you get when you *download* the data tar file. Subsequently, you may obtain additional needed data files from the *NCAR Community Data Portal*. Here is a partial schematic of the datadir (where "tiegcmx.xx" is the desired model version):

```
                datadir
                   |
---------------------------------------------
          |                        |
     source *.nc              tiegcmx.xx
     gswm *.nc                    |
     imf *.nc                seasons/*.nc
        etc               climatology/*.nc
                             dec2006/*.nc
                                 etc
```

Here, source *.nc refers to *SOURCE* start-up files, gswm *.nc refers to *GSWM data files*, and imf *.nc refers to *IMF_NCFILE* data files.

"tiegcmx.xx" refers to the version of the model that was downloaded. This subdirectory contains history file output from test/benchmark runs executed by that version of the model (see *test/* subdirectory of *modeldir*).

RUN CONTROL PARAMETERS: THE NAMELIST INPUT FILE

The *namelist input* file specifies user-provided run control parameters for a model run. These parameters include the model startup file, start and stop times, solar inputs, lower boundary files, and several other flags and data files as necessary. This document describes each valid namelist parameter, their valid combinations, and provides several example input files for running the model in different modes and resolutions.

4.1 Example Namelist Input Files

Please refer to the following examples of namelist input files:

Note: Any part of a line in the namelist file following a semi-colon will be treated as a comment (see example files).

Example Namelist Input Files:

- The default input file: `default.inp`

- A continuation of the default run: `continuation.inp`

- Saving diagnostics on secondary history files: `diags.inp`

- Full year climatology with constant forcing: `climatology.inp`

- Seasonal control runs with constant forcing: `mareqx.inp`

- Heelis potential model with constant solar forcing: `heelis_smax.inp`

- Heelis potential model with GPI (Kp) data file: `heelis_gpi.inp`

- Weimer potential model with constant IMF forcing: `weimer_con.inp`

- Weimer potential model with IMF data file: `weimer_imf.inp`

- Weimer potential model with IMF+GPI data files: `weimer_imf+gpi.inp`

- Default double-resolution run:

- Example "storm runs":

    - December, 2006 "AGU" storm run: `dec2006.inp`

4.2 Explanation of Valid Namelist Parameters

Following is a table of valid TIEGCM 1.94 namelist parameters, and their descriptions. Follow the parameter name links to explanations below.

Parameter Name	Data Type and Default	Description
AURORA	integer: 1	0/1 flag for auroral parameterization
BXIMF or BXIMF_TIME	real or real array	X-component of the IMF
BYIMF or BYIMF_TIME	real or real array	Y-component of the IMF
BZIMF or BZIMF_TIME	real or real array	Z-component of the IMF
CALENDAR_ADVANCE	real: 1	0/1 switch to advance calendar time
COLFAC	real: 1.5	O-O+ collision factor
CTPOTEN	real:	Cross-Tail Potential
CTPOTEN_TIME	real: [none]	Time-dependent Cross-Tail Potential
F107 or F107_TIME	real or real array	Daily F10.7 cm solar flux
F107A or F107A_TIME	real or real array	81-day average F10.7 cm solar flux
GPI_NCFILE	string: [none]	Geophysical Indices (Kp) data file
GSWM data files	string: [none]	GSWM Model tidal lbc data files
HIST	integer(3)	Primary history write frequency
KP or KP_TIME	real or real array	Kp for calc of hpower and ctpoten
LABEL	string:	Arbitrary string identifying the run
MXHIST_PRIM	integer: 10	Max histories on primary file
MXHIST_SECH	integer: 24	Max histories on secondary file
OUTPUT	string array	Primary history output file(s)
POTENTIAL_MODEL	string: [HEELIS]	High-latitude Potential Model
POWER or POWER_TIME	real or real array	Hemispheric Power (GW)
SECHIST	integer(3)	Secondary history write frequency
SECFLDS	string array	Fields to be stored on secondary histories
SECOUT	string array	Secondary history output file(s)
SOURCE	string: [none]	Primary SOURCE (start-up) file
SOURCE_START	integer(3)	Model time to start on SOURCE file
START	integer(3)	Model start time (day,hour,minute)
START_YEAR	integer: 2002	Starting year
START_DAY	integer: 80	Starting day of year
STEP	integer: [none]	Model time step (seconds)
STOP	integer(3)	Model stop time (day,hour,minute)
SWDEN or SWDEN_TIME	real or real array	Solar Wind Density
SWVEL or SWVEL_TIME	real or real array	Solar Wind Velocity
TIDE	real(10)	Amplitudes and phases of semi-diurnal tide
TIDE2	real(2)	Amplitudes and phases of diurnal tide

AURORA

If AURORA > 0 then the auroral parameterization (aurora.F) is called by dynamics (dynamics.F), otherwise it is not called.

Data type: scalar integer

Default: 1

Back to top

BXIMF or BXIMF_TIME

X-component of the IMF. Can be specified as either a constant (BXIMF), or series of time-dependent values (BXIMF_TIME). If IMF_NCFILE is set and BXIMF is not provided, then BXIMF will be taken from the IMF data file.

Data type: real or real array

Examples:

- BXIMF = 0. ; constant for entire run

- BXIMF_TIME = 80,0,0,40., 80,1,0,30., 80,5,0,20. ; time series

See also:

- *BYIMF or BYIMF_TIME*

- *BZIMF or BZIMF_TIME*

- *IMF_NCFILE*

Back to top

BYIMF or BYIMF_TIME

Y-component of the IMF. Can be specified as either a constant (BYIMF), or series of time-dependent values (BYIMF_TIME). If IMF_NCFILE is set and BYIMF is not provided, then BYIMF will be taken from the IMF data file.

Data type: real or real array

Examples:

- BYIMF = 0. ; constant for entire run

- BYIMF_TIME = 80,0,0,40., 80,1,0,30., 80,5,0,20. ; time series

See also:

- *BXIMF or BYIMF_TIME*

- *BZIMF or BZIMF_TIME*

- *IMF_NCFILE*

Back to top

BZIMF or BZIMF_TIME

Z-component of the IMF. Can be specified as either a constant (BZIMF), or series of time-dependent values (BZIMF_TIME). If IMF_NCFILE is set and BZIMF is not provided, then BZIMF will be taken from the IMF data file.

Data type: real or real array

Examples:

- BZIMF = 0. ; constant for entire run

- BZIMF_TIME = 80,0,0,40., 80,1,0,30., 80,5,0,20. ; time series

See also:

- *BXIMF or BXIMF_TIME*

- *BYIMF or BYIMF_TIME*

- *IMF_NCFILE*

Back to top

4.2. Explanation of Valid Namelist Parameters

CALENDAR_ADVANCE

Set CALENDAR_ADVANCE=1 to advance calendar time from START_DAY, otherwise calendar time is not advanced. If advancing calendar time, iday (init_module) is incremented every 24 hours, and the sun's declination and longitude is recalculated (see sub advance_day in advance.F and sub sunloc in magfield.F), thereby allowing seasonal change to take place. The earth's orbital eccentricity "sfeps" is also updated as a 6% variation in solar output over a year.

A run with CALENDAR_ADVANCE=0 is referred to as a "steady-state" run. This is often used to advance the model to a "steady-state" for a given date, prior to a seasonal run with CALENDAR_ADVANCE=1.

Back to top

COLFAC

O-O+ Collision Frequency, alias the "Burnside Factor". Default is 1.5, but there have been recommendations for 1.3. COLFAC is used in lamdas.F and oplus.F.

Data type: real
Default: 1.5

Back to top

CTPOTEN or CTPOTEN_TIME

Cross-tail (or cross-cap) potential. This is used in the auroral precipitation parameterization. It can be provided either as a single constant (CTPOTEN), or several time-dependent values (CTPOTEN_TIME). If GPI_NCFILE is set and CTPOTEN is not provided, it will be calculated from 3-hourly Kp data read from GPI_NCFILE.

The time-dependent example below specifies increasing CTPOTEN from model times 80,0,0 to 80,1,0, and 80,5,0. Interpolated values will be used between these specified model times.

Note that if POTENTIAL_MODEL='WEIMER' or 'WEIMER05', then the user is not allowed to provide CT-POTEN because it will be calculated from the Weimer electric potential.

Data type: real or real array

Examples:

- CTPOTEN = 60.

- CTPOTEN_TIME = 80,0,0,60., 80,1,0,65., 80,5,0,100.

See also:

- *POWER or POWER_TIME*

- *KP or KP_TIME*

- *GPI_NCFILE*

Back to top

F107 or F107_TIME

Daily F10.7 cm solar flux. This can be provided either as a single constant (F107), or several time-dependent values (F107_TIME). If GPI_NCFILE is set and F107 is not set, then F107 will be set from the data. The below example of F107_TIME increases the f10.7 flux from 120 to 150 in the first hour of model time, then to 200 by the fifth hour. Values are linearly interpolated at each time-step.

Data type: real or real array

Examples:

- F107 = 120.

- F107_TIME = 80,0,0,120., 80,1,0,150., 80,5,0,200.

See also:

- *F107A*

- *POTENTIAL_MODEL*

- *GPI_NCFILE*

- *IMF_NCFILE*

Back to top

F107A or F107A_TIME

81-day average F10.7 cm solar flux. This can be provided either as a single constant (F107A), or several time-dependent values (F107A_TIME). If GPI_NCFILE is set and F107A is not set, then F107A will be set from the data. The below example of F107A_TIME increases the f10.7a flux from 120 to 130 in 12 hours of model time.

Data type: real or real array

Examples:

- F107A = 120.

- F107A_TIME = 80,0,0,120., 80,6,0,125., 80,12,0,130.

See also:

- *F107*

- *POTENTIAL_MODEL*

- *GPI_NCFILE*

- *IMF_NCFILE*

Back to top

GPI_NCFILE

Specifies a netCDF data file containing 3-hourly Kp and daily F10.7 data to drive high-latitude convection and the auroral precipitation oval. If GPI_NCFILE is specified, and POTENTIAL_MODEL='HEELIS', then at least one of CTPOTEN,POWER,F107,F107A must **not** be specified. If CTPOTEN or POWER are not specified, they are calculated from the Kp data using empirical relationships (see source file gpi.F). If F107 or F107A are not specified, the data will be used.

If GPI_NCFILE is specified when POTENTIAL_MODEL='WEIMER' and IMF_NCFILE is specified, then the Weimer model and aurora will be driven by the IMF data, and only F107 and F107A will be read from the GPI data file (F107 is not available on IMF data files).

If the current model time is not available on the GPI data file, the model will print an error message to stdout, and stop.

Data Source: Ascii data is obtained from NOAA/NGDC, and an equivalent netCDF data file is written for import to the TGCM models (see code in hao:$TGCMROOT/mkgpi).

Datatype: string

Example:

- GPI_NCFILE = '$TGCMDATA/gpi_2000001-2009031.nc'

See also:

- *CTPOTEN or CTPOTEN_TIME*

- *POWER or POWER_TIME*

- *F107 or F107_TIME*

- *IMF_NCFILE*

Back to top

GSWM model data files for lbc

Paths to netCDF data files containing tidal perturbations from the Global Scale Wave Model. If provided, the files will be read and the perturbations will be added to the lower boundary conditions of T,U,V, and Z. If provided, then TIDE and TIDE2 must be zeroed out.

Warning: As of version 1.94, the model is not tuned to use the non-migrating GSWM tidal components. The default namelist input file specifies migrating diurnal and semi-diurnal tides, but not the non-migrating components. In later releases, non-migrating tides may be supported at the 2.5-deg resolution.

GSWM files must contain data compatable with the lower boundary of the model (99 km), and the horizontal resolution of the model being run (either 5 or 2.5 degrees). See examples below.

Datatype: string

Examples:

- GSWM files for the 5-degree TIEGCM:

```
GSWM_MI_DI_NCFILE   = '$TGCMDATA/gswm_diurn_5.0d_99km.nc'
GSWM_MI_SDI_NCFILE  = '$TGCMDATA/gswm_semi_5.0d_99km.nc'
GSWM_NMI_DI_NCFILE  = '$TGCMDATA/gswm_nonmig_diurn_5.0d_99km.nc'
GSWM_NMI_SDI_NCFILE = '$TGCMDATA/gswm_nonmig_semi_5.0d_99km.nc'
```

- GSWM files for 2.5-degree TIEGCM:

```
GSWM_MI_DI_NCFILE   = '$TGCMDATA/gswm_diurn_2.5d_99km.nc'
GSWM_MI_SDI_NCFILE  = '$TGCMDATA/gswm_semi_2.5d_99km.nc'
GSWM_NMI_DI_NCFILE  = '$TGCMDATA/gswm_nonmig_diurn_2.5d_99km.nc'
GSWM_NMI_SDI_NCFILE = '$TGCMDATA/gswm_nonmig_semi_2.5d_99km.nc'
```

See also:

- *TIDE*

- *TIDE2*

Back to top

HIST

Primary history write frequency, specified as a model time (day,hour,minute). HIST time must divide evenly into STOP minus START times.

Examples:

- HIST = 1,0,0 ;request daily histories

- HIST = 0,1,0 ;request hourly histories

- HIST = 0,0,12 ;request 12-minute histories

See also:

- *SECHIST*

Back to top

POWER or POWER_TIME

Hemispheric Power (GW). This is used in the auroral precipitation parameterization. It can be provided either as a single constant (POWER), or several time-dependent values (POWER_TIME). If GPI_NCFILE is set and POWER is not provided, it will be calculated from 3-hourly Kp data read from GPI_NCFILE.

The time-dependent example below specifies increasing POWER from model times 80,0,0 to 80,1,0, and 80,5,0. Interpolated values will be used between these specified model times.

Data type: real or real array

Examples:

- POWER = 16.

- POWER_TIME = 80,0,0,16., 80,1,0,20., 80,5,0,70.

See also:

- *CTPOTEN or CTPOTEN_TIME*

- *KP or KP_TIME*

- *GPI_NCFILE*

Back to top

KP or KP_TIME

Geomagnetic Activity index. If KP is specified and POWER and/or CTPOTEN are commented, then the given KP will be used with empirical formulas to calculate POWER and/or CTPOTEN, which are used in the Auroral parameterization.

KP can be provided as a scalar constant (KP), or as a series of time-dependent values (KP_TIME), as in the below examples. KP cannot be set if GPI_NCFILE data file is specified.

Empirical formula used to calculate POWER from KP (see function hp_from_kp in util.F):

```
if (kp <=7.) hp_from_kp = 16.82*exp(0.32*kp)-4.86
if (kp > 7.) hp_from_kp = 153.13 + (kp-7.)/(9.-7.)*(300.-153.13)
```

Empirical formula used to calculate CTPOTEN from KP (see function ctpoten_from_kp in util.F):

```
ctpoten_from_kp = 15.+15.*kp + 0.8*kp**2
```

Examples:

- KP = 4.0

- KP_TIME = 80,0,0,4., 80,6,0,4.5, 80,12,0,5.0

See also:

- *CTPOTEN*

- *POWER*

- *GPI_NCFILE*

Back to top

IMF_NCFILE

Specifies a netCDF data file containing hourly IMF parameters BX,BY,BZ,SWVEL, and SWDEN. This can be set only when POTENTIAL_MODEL='WEIMER'. The data will be used to drive the Weimer 2005 potential model. When set, the user must **not** provide at least one of the above IMF parameters. Data will be used for IMF parameters not provided by the user. Values (scalar or time-dependent) that are provided by the user will take precedence over the data file.

If the current model time is not available on the IMF data file, the model will print an error message to stdout and stop.

Ascii data is obtained from NASA/OMNI (Combined 1AU IP Data), and an equivalent netCDF data file is written for import to the model.

Example:

- IMF_NCFILE = '$TGCMDATA/imf_OMNI_2002001-2002365.nc'

Back to top

LABEL

LABEL may be any string up to 80 characters long, used to identify a run. The LABEL is written to output history files as a global file attribute. This parameter is purely a user convenience, and does not effect the model run in any way.

Data type: string
Default: 'tiegcm res=5'

Back to top

MXHIST_PRIM

Maximum number of histories to be written to primary OUTPUT files. When this many histories have been written to the current OUTPUT file, the next OUTPUT file is created and it receives subsequent histories. This parameter can be adjusted to control the size of primary OUTPUT files.

Data type: integer
Default: 10

Examples:

- MXHIST_PRIM = 15 ; allow maximum of 15 histories per primary output file

See also:

- *OUTPUT*

Back to top

MXHIST_SECH

Maximum number of histories to be written to secondary output files (SECOUT). When this many histories have been written to the current SECOUT file, the next SECOUT file is created and it receives subsequent histories. This parameter can be adjusted to control the size of secondary OUTPUT files.

Data type: integer
Default: 24

Examples:

- MXHIST_SECH = 24 ; allow 1 day of hourly histories per file
- MXHIST_SECH = 48 ; allow 2 days of hourly histories per file

See also:

- *SECOUT*

Back to top

OUTPUT

List of primary history output files. Each file may be an absolute path, or relative to the execution directory. If an initial run (SOURCE is specified), then pre-existing OUTPUT files will be overwritten. If a continuation run (SOURCE is *not* specified), then the first OUTPUT file should contain the source history at START time. In this case, subsequent output histories will be appended to the first OUTPUT file until it is full. As each OUTPUT file is filled (see MXHIST_PRIM), the next OUTPUT file is created and histories are written until it is full, and so on.

OUTPUT files are usually specified with increasing integers imbedded in the names. See examples below. As a convenience, large sequences of files may be specified in a "short-form", see example 3 below specifying 20 files. By convention, primary history output files may use the letter "p" to indicate primary file series (see all 3 examples below, and contrast with SECOUT).

Examples:

```
OUTPUT = 'p_myoutput_001.nc'
OUTPUT = 'myoutput.p001.nc','myoutput.p002.nc','myoutput.p003.nc'
OUTPUT = 'myoutput_p001.nc','to','myoutput_p020.nc','by','1'
```

See also:

- *SECOUT*

- *SOURCE*

- *MXHIST_PRIM*

Back to top

POTENTIAL_MODEL

The high-latitude potential model used to calculate electric potential above a specified latitude. This string can have one of two values:

POTENTIAL_MODEL = 'HEELIS'
POTENTIAL_MODEL = 'WEIMER'

'HEELIS' is the Rod Heelis model (heelis.F). 'WEIMER' is the Dan Weimer 2005 model (wei05sc.F).

Data type: string
Default: 'HEELIS'

Back to top

SECFLDS

List of fields to be saved to secondary histories. These may be either fields that are also saved on primary histories (so-called "prognostic" fields), fields that have been requested via addfld calls in the source code, or fields available via the diagnostics module (see example below).

Note the final size of secondary output files is affected by the number of fields specified as well as the number of histories on the file. The file size can be controlled by setting the number of histories allowed on a secondary file *MXHIST_SECH*.

Data type: one or more character strings

Examples:

```
;
; Example for tiegcm1.9: all fields are "prognostic" except EEX,EEY,EEZ,
; which are saved by addfld calls in sub ionvel (ionvel.F).
;
 SECFLDS = 'TN','UN','VN','O1','NO','N4S','NE','TE','TI',
           'O2','O2P','OMEGA','POTEN','EEX','EEY','EEZ'
;
; This example lists all diagnostic fields available via the diags module
; (it is not necessary to call addfld in the code to obtain these fields)
;
 SECFLDS = 'CO2_COOL','NO_COOL','DEN','HEATING','QJOULE',
           'SIGMA_PED','SIGMA_HAL','TEC','UI_ExB','VI_ExB','WI_ExB',
           'LAMDA_PED','LAMDA_HAL','HMF2','NMF2','SCHT','MU_M'
```

See also: *MXHIST_SECH*

Back to top

SECHIST

Secondary history write frequency, specified as a model time (day,hour,minute). SECHIST time must divide evenly into SECSTOP minus SECSTART times.

Data type: 3 integers (day,hour,minute)
Valid range: 0-365 for day, 0-23 for hour, 0-59 for minute.

Examples:

- SECHIST = 0,1,0 ;request hourly histories

- SECHIST = 0,0,12 ;request 12-minute histories

See also:

- *HIST*

Back to top

SECOUT

List of secondary history output files. Secondary histories store diagnostic fields, usually at a higher temporal resolution than primary files. Each file may be an absolute path, or relative to the execution directory. Beware that SECOUT will overwrite any pre-existing files with the same names. As each SECOUT file is filled (see MXHIST_SECH), the next SECOUT file is created and histories are written until it is full, and so on.

SECOUT files are usually specified with increasing integers imbedded in the names. See examples below. As a convenience, large sequences of files may be specified in a "short-form", see example 3 below specifying 20 files. By convention, secondary history output files may use the letter "s" to indicate secondary file series (see all 3 examples below).

Examples:

```
SECOUT = 's_myoutput_001.nc'
SECOUT = 'myoutput.s001.nc','myoutput.s002.nc','myoutput.s003.nc'
SECOUT = 'myoutput_s001.nc','to','myoutput_s020.nc','by','1'
```

See also:

- *OUTPUT*

- *SOURCE*

- *MXHIST_SECH*

Back to top

SOURCE

SOURCE is the start-up history file for an initial run. SOURCE may be a full path or relative to the execution directory. It must be a TIEGCM history with the same grid resolution as the model being run. It does not need to be from the same model version as that being run.

If SOURCE is specified, then SOURCE_START, the model time of the history to read on the SOURCE file, must also be specified. The code will search the SOURCE file for the SOURCE_START history. If SOURCE is *not* specified, then the run is a continuation run, and the source history is provided in the first OUTPUT file at START time.

The SOURCE file must be on the local disk. The model will not look for the SOURCE history on any archive file system.

Examples:

- SOURCE = '$TGCMDATA/TGCM.tiegcm1.94.pcntr_eqnx_smed.nc'

See also:

- *SOURCE_START*

Back to top

SOURCE_START

This is the model time of the history to read from the SOURCE file. Model time is specified as a 3-integer triplet: day,hour,minute. If SOURCE is specified, then SOURCE_START must also be specified. If the SOURCE_START history is not found on the SOURCE file, the model will stop and print an appropriate error message to stdout.

Data type: 3 integers
Valid range: 0-365 for day, 0-23 for hour, 0-59 for minute.

Example:

- SOURCE_START = 80,0,0

See also:

- *SOURCE*

Back to top

START

Model time for start of the run. Model time is a 3-integer triplet: day,hour, minute. If CALEN-DAR_ADVANCE=0, then START day can be any number between 0 and 365. If CALENDAR_ADVANCE=1,

then START day must be the same as START_DAY. If an initial run, START time does not have to be the same as SOURCE_START.

Data type: 3 integers Valid range: 0-365 for day, 0-23 for hour, 0-59 for minute.

Examples:

- START = 80,0,0

See also:

- *SOURCE_START*

Back to top

START_DAY
Calendar starting day.

Data type: integer
Default: 80
Valid range: 1 to 365

Back to top

START_YEAR
Starting year for the run.

Data type: integer
Default: 2002

Back to top

STEP
Model time-step in seconds. Default value is 120, although during periods of quiet solar activity, the model will run fine at 180. During periods of intense solar activity (e.g., F10.7 > 200, or high magnitude BZ southward), the model may become numerically unstable. In this case, reducing the timestep to as low as 60 seconds may help the model get through the rough period.

Data type: integer
Default: Usually 120 or 180

Back to top

STOP
Model stop time for the run. Model time is specified as a 3-integer triplet: day,hour,minute.

Data type: 3 integers
Valid range: 0-365 for day, 0-23 for hour, 0-59 for minute.

Example:

- STOP = 81,0,0

Back to top

SWDEN or SWDEN_TIME

Solar Wind Density. Can be specified as either a constant (SWDEN), or series of time-dependent values (SWDEN_TIME). If IMF_NCFILE is set and SWDEN is not provided, then it SWDEN will be taken from the IMF data file.

Data type: real or real array

Examples:

- SWDEN = 4.0 ; constant for entire run

- SWDEN_TIME = 80,0,0,2., 80,1,0,3., 80,2,0,4. ; time series

See also:

- *IMF_NCFILE*

Back to top

SWVEL or SWVEL_TIME

Solar Wind Velocity. Can be specified as either a constant (SWVEL), or series of time-dependent values (SWVEL_TIME). If IMF_NCFILE is set and SWVEL is not provided, then it SWVEL will be taken from the IMF data file.

Data type: real or real array

Examples:

- SWVEL = 400. ; constant for entire run

- SWVEL_TIME = 80,0,0,100., 80,1,0,200., 80,2,0,300. ; time series

See also:

- *IMF_NCFILE*

Back to top

TIDE

Hough mode amplitudes and phases of the semi-diurnal tide. If GSWM tidal perturbations are specified, TIDE should be set to 0.

Data type: 10 reals

Example:

```
TIDE= 1.9300E+04, 1.5000E+04, 2.3100E+04, 0.7700E+04, 0.1660E+04,
      -2.600E+00,  0.000E+00, -3.300E+00, 4.2000E+00, 5.0000E+00
```

See also:

- *GSWM_MI_SDI_NCFILE*

- *GSWM_NM_SDI_NCFILE*

Back to top

TIDE2

Hough mode amplitudes and phases of the diurnal tide. If GSWM tidal perturbations are specified, TIDE2 should be set to 0.

Data type: 2 floats

Example:

```
TIDE2 = 4.1E+4, -3.7
```

See also:

- *GSWM_MI_DI_NCFILE*
- *GSWM_NM_DI_NCFILE*

Back to top

STRUCTURE OF OUTPUT HISTORY FILES

5.1 NetCDF Output Files (*netCDF*)

TIEGCM history files are output in netCDF, a self-describing platform-independent data format written and maintained by the UCAR Unidata program.

Each netCDF file contains one or more *histories*, i.e., the state of the model output fields at a discrete instant in *model time*. Here is an example of the metadata content of a sample primary history file: `primary.ncd`. This example file contains six daily histories (days 355 to 360 of 2002). This metadata is obtained via the "ncdump" command in the netCDF utilities. This example ncdump file contains data values for scalar and singly-dimensioned vectors only.

TIEGCM history files are "CF compliant", i.e., they conform to the NetCDF Climate and Forecast (CF) Metadata Convention.

5.2 Primary and Secondary History Files

"Primary" history files contain the "prognostic" fields necessary to restart the model. They can be specified in namelist input as the *SOURCE* file for starting the model in an *initial run* (a *continuation run* does not specify a SOURCE file, and is continued from the START time found on the first OUTPUT file). Typically, daily histories are stored on primary history files.

Fields on primary histories necessary for start-up of the TIEGCM are as follows: TN, UN, VN, O2, O1, N4S, NO, O+, N2D, TI, TE, NE, O2P, OMEGA, Z, POTEN

"Secondary" history files contain diagnostic fields and/or primary history fields. Fields to be saved on the secondary history files are listed by the namelist input parameter *SECFLDS*. Diagnostics can be saved by calling subroutine addfld in the code (addfld.F), or by including one or more of the "standard" diagnostic fields available via the *diagnostics* module (diags.F). Secondary histories are often saved at a higher temporal resolution than primary histories, typically hourly. `Here` is an ncdump of an example secondary history file.

SAVING DIAGNOSTIC FIELDS

The diagnostics module (diags.F) in the TIEGCM will calculate and save diagnostic fields to the secondary histories. The user can add any subset of these fields to the *SECFLDS* parameter list in the namelist input file. See the `diagnostics namelist example`.

6.1 Table of Available Diagnostics

A table of available diagnostic fields for TIEGCM version 1.94, is `here`. To request that selected fields be saved to the secondary histories, add the "Shortname" of the desired fields to the *SECFLDS* parameter list in the namelist input file.

Columns of the diagnostics table are described as follows:

- Field: Field number (not significant)

- Shortname: Short name of the field (use these in the namelist SECFLDS list)

- Units: Units of the field (most of these are udunits compliant)

- Levels: The cell position in the vertical dimension on which the field was calculated:

    - lev: Field was calculated on midpoints (geographic grid)

    - ilev: Field was calculated on interfaces (geographic grid)

    - mlev: Field was calculated on midpoints (magnetic grid)

    - imlev: Field was calculated on interfaces (magnetic grid)

    - none: Field is 2d (lat x lon), so no vertical component

- Caller: The function or subroutine from which the diags module is called.

- Longname: Long name of the field.

A complete list of diagnostics for the SECFLDS namelist parameter is included at the end of the diags table.

6.2 Saving Fields/Arrays from the Source

Arbitrary 2d and 3d arrays can be saved from the model to secondary histories by inserting a call to subroutine addfld (addfld.F) in the source code. There are many examples of this in the source code, just grep on "call addfld". For more information about how to make calls to addfld, please see comments in the addfld.F source file.

Here are a couple of examples of addfld calls from near the end of subroutine qrj (qrj.F). These calls are inside a latitude loop, where the loop variable index is "lat". Normally, in parallel code, subdomains of the field are passed, e.g., lon0:lon1 and lat0:lat1:

```
 call addfld('QO2P'  ,' ',' ',  qo2p(lev0:lev1,lon0:lon1,lat),
|  'lev',lev0,lev1,'lon',lon0,lon1,lat)
 call addfld('QN2P'  ,' ',' ',  qn2p(lev0:lev1,lon0:lon1,lat),
|  'lev',lev0,lev1,'lon',lon0,lon1,lat)
 call addfld('QNP'   ,' ',' ',   qnp(lev0:lev1,lon0:lon1,lat),
|  'lev',lev0,lev1,'lon',lon0,lon1,lat)
```

The calling sequence for subroutine addfld is explained in comments at the `top of addfld.F`.

MODEL SOURCE CODE

The source code is in the *src/* subdirectory of the model root directory (*modeldir*), which is provided in the model *download* file.

7.1 The Academic License Agreement

The TIEGCM `Open Source Academic Research License Agreement` specifies the terms and restrictions under which the NCAR/UCAR grants permission to use the model, including the source code, for research, academic, and non-profit purposes.

7.2 Source Code Flow Diagram

A detailed flow diagram and calling tree of the source code structure is available in single and multi-page pdf files:

> **Warning:** Some areas of these flow charts are out of date with respect to TIEGCM version 1.94

- `TIEGCM Code Structure (multi-page pdf)`
- `TIEGCM Code Structure (single-page pdf)`

7.3 Modifying the Source Code

As a community user, student, research scientist or developer, you may need to modify the model source code. It is best to do this after building and at least making a default execution of the model (see the *QuickStart* Section). To change one or more source files, simply go to the *src/* subdirectory in the model root directory *modeldir*, and edit the files as necessary. Then return to the working directory *workdir* and re-execute the job script. It will recompile the modified files, and any other source files that depend on the modified files, and re-execute the model. Alternatively, you can enter the execution directory *execdir*, and recompile the code by typing "gmake" on the command line, then return to the working directory and re-execute the job script.

THE MAKE/BUILD PROCESS: COMPILE AND LINK

The TIEGCM model is formally supported on two platforms systems: 64-bit Linux, and IBM/AIX. However, the model has been built and executed on several other platforms. The source code is f90 standard compliant, and is mostly fixed-format fortran.

Compilers used on Linux systems include Intel ifort, and PGI's pgf90. The compiler used on the NCAR IBM/AIX system is xlf90.

Library dependencies consist mainly of netCDF and MPI. The MPI library is often bundled in with the compiler. Locations of these libraries are specified in "Make.machine" files, which set platform-specific compile flags and other parameters for the build process. The following Make.machine files are provided in the *scripts/* directory:

- `Make.bluefire` (NCAR IBM/AIX machine)

- `Make.intel_hao64` (ifort compiler on HAO 64-bit Linux desktops)

- `Make.pgi_hao32` (pgf90 compiler on HAO 32-bit Linux desktops)

- `Make.pgi_hao64` (pgf90 compiler on HAO 64-bit Linux desktops)

One of these files, or the user's own, is specified by the csh variable "make" in the *job script*. The specified file is included in the main `Makefile`.

VERSION BENCHMARK TEST RUNS

A series of benchmark runs are made for each version release of the TIEGCM. Job scripts and namelist input files for these runs are available in the "tests" directory under the model root directory.

For version 1.94, the following runs were made:

- Steady-state seasonal runs (start-up histories and 5-day control runs) at solar minimum and solar maximum conditions:

    - March Equinox (day 80)

    - June Solstice (day 172)

    - September Equinox (day 264)

    - December Solstice (day 355)

- Full-year Climatology:

    - Heelis potential model with constant solar forcing

- December, 2006 "AGU" storm case:

    - Heelis potential model with GPI (Kp) data

    - Weimer potential model with IMF data (F10.7 from GPI)

TROUBLE SHOOTING

10.1 Problems Downloading the Model and Data

10.2 Problems with the Build Process

10.3 Namelist Input Problems

10.4 Debugging Model Runs

10.5 Platform Porting

10.6 Performance Issues

10.7 Contact Model Support Group at HAO

GLOSSARY

continuation run A continuation run continues from the last history of the previous run. That history (*START* time) must be on the first *OUTPUT* file provided by the namelist input file. A continuation run must not specify a *SOURCE* file or *SOURCE_START* time.

datadir The directory containing start-up and other input data files required for running the model. A minimal set of datadir files are available via *download*. The datadir is sometimes referred to by the *TGCMDATA* environment variable. Additional data files are available via the *NCAR Community Data Portal*.

doc/ Subdirectory under the *modeldir* containing documentation, e.g., the User's Guide, Model Description, Release Notes, etc.

execdir The model execution directory. This is the directory where the model is built and executed. It is typically, but not necessarilly, a subdirectory of your working directory *workdir*. When a job script is executed from a working directory, the execdir is created if it does not already exist. During a model run, output history files are written to the execdir.

history A model history records the state of the model at a discrete instant in *model time*. One or more histories are stored in netCDF history files.

initial run An initial run is started from a history on a *SOURCE* file (see also *SOURCE_START*). Subsequent *continuation runs* do not provide SOURCE or SOURCE_START, but rather search for the *START* time on the first *OUTPUT* history file provided in the namelist input, and continue the run from there.

model time TIEGCM model time is represented by an integer triplet: day,hour,minute, where day is the julian day of the year, and hour is the ut. The variable for model time on history files is mtime(3,ntimes). For example, a history file may contain 24 hourly histories for day 80: mtime = 80,1,0, 80,2,0, ... 81,0,0.

namelist input The model reads user specified parameters from the *namelist input file* via f90 standard namelist read. Keyword/Value pairs are read from unit 5, and are validated by the input module (input.F).

NCAR Community Data Portal The NCAR Community Data Portal is a public data repository with datasets from NCAR, UCAR, UOP, and participating organizations. To browse TIEGCM-related files (mostly netCDF history files for model start-up, or results of benchmark runs), click on the "Models" link, then to the "Thermospheric General Circulation Models" link, and finally to the desired model version. NetCDF Metadata is available without actually downloading files.

netCDF TIEGCM output history files are written in netCDF, a self-describing platform-independent data format written and maintained by the UCAR Unidata program.

resolution The TIEGCM can be run in one of two resolutions:

- 5 x 5 deg lat x lon, 2 grid levels per scale height (dz = 0.50)

- 2.5 x 2.5 deg lat x lon, 4 grid levels per scale height (dz = 0.25)

The resolution is set by the "modelres" shell variable in the TIEGCM *job script*.

Note: The 2.5-degree resolution model is available in version 1.94, but it is not fully validated or supported by the public release.

---

scripts/ Subdirectory under the *modeldir* containing supporting and utility scripts, including job scripts, the default namelist input file, several Make files, etc.

src/ Subdirectory under the *modeldir* containing the model source code (*.F, *.h files).

test/ Subdirectory under the *modeldir* containing setup scripts for benchmark runs.

TGCMDATA A unix environment variable that refers to the *datadir*. This environment variable may be used when referring to data files in the namelist read file, e.g., "GPI_NCFILE = *$TGCMDATA/gpi_xxxxx.nc*". See *namelist read files*.

modeldir The model root directory. This directory typically contains subdirectories *src/* (model source code), *scripts/* (utility scripts), *doc/* (documentation), and *test/* (test runs). The modeldir is available via *download*, and is typically a subdirectory of the model working directory (*workdir*).

workdir Your local working directory. This will typically contain the model root directory *modeldir*, the execution directory *execdir*, and related namelist input files, job scripts, stdout files, etc. It may also contain a data subdirectory *datadir*.

INDICES AND TABLES

- *genindex*
- *search*

Note: This document was last updated on May 05, 2011

INDEX